2021年度ABB杯智能技术创新大赛

——俄罗斯方块玩家挑战赛

目　录

1 赛题重述及分析

1.1 赛题重述

1.1.1 初赛赛题场景设定

将现实场景中的码垛应用对应于俄罗斯方块游戏，增加趣味性，同时考察参赛选手工具的学习使用能力，抓取路径的规划和优化，叠放方案的规划和优化，以及对应算法逻辑的缜密性。

(a) 设定型号的机械臂A，及设定的机械臂运动速度；

(b1) 俄罗斯方块的放置平台B，该平台每次会生成一个形状类型随机的俄罗斯方块；

(c) 俄罗斯方块仓库C，给定固定的长*宽*高（1005mm* 105mm* 2000mm），其中宽度约为一个俄罗斯方块的最小单元的宽度；

(d) 其中，场景中的A、B、C相对位置固定；俄罗斯方块D的模型，包括尺寸、位姿、类型，按照传统游戏规则，每一个俄罗斯方块的最小单元为一个正方体unit；(详见图1.1、图1.2)；

(e) 给定一个固定的随机规则：要求参赛选手使用.net framework里的RNGCryptoServiceProvider来生成方块标识随机数，随机量包括俄罗斯方块的类型，位置和姿态信息，详见此处；

(f) 俄罗斯方块模型文件可供下载，详见此处。

1.1.2 决赛赛题场景设定

将上述b1的条件进行修改为b2, 其他保持不变。主要考察参赛选手对整体俄罗斯方块游戏的解决方案的设计及优化，关注全局性：

(b2) ABB现场提供俄罗斯方块列表，供选手程序使用；或者提供一筐给定的俄罗斯方块模型（选手在设计时需考虑列表的读取和相应的数据结构）。决赛赛题详情将于9月1日公布。

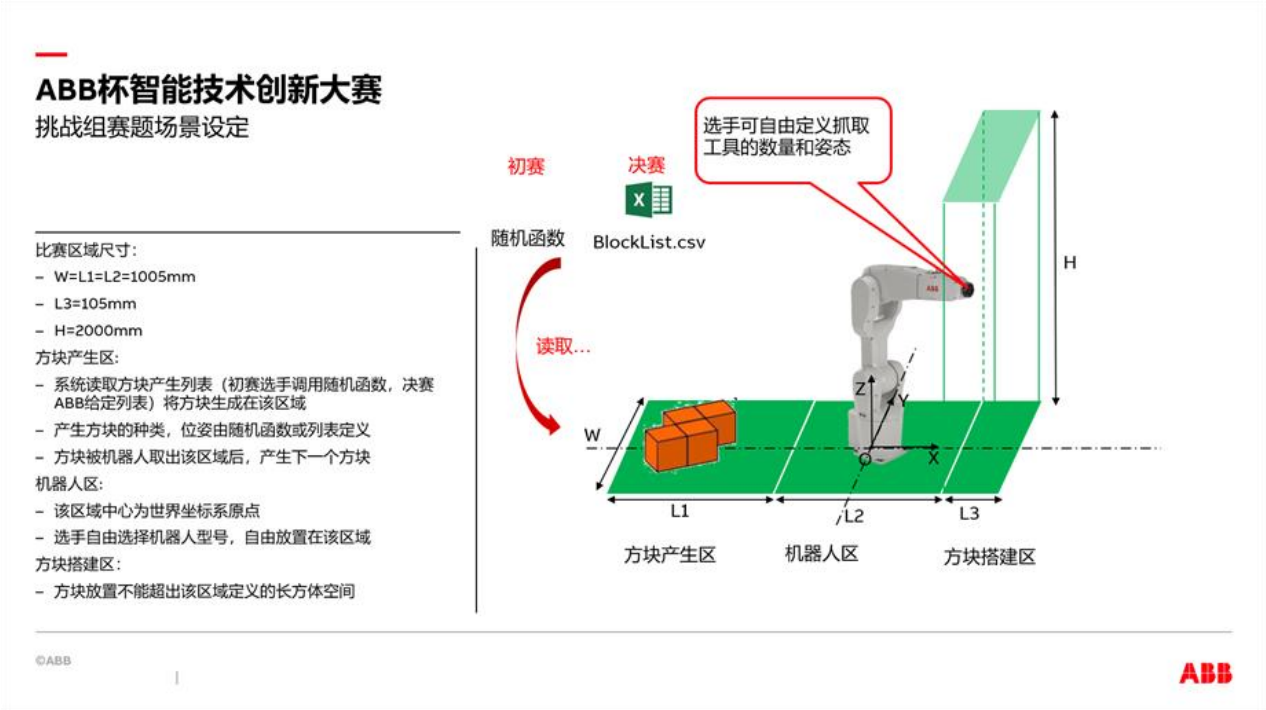


图1.1 挑战组赛题场景设定

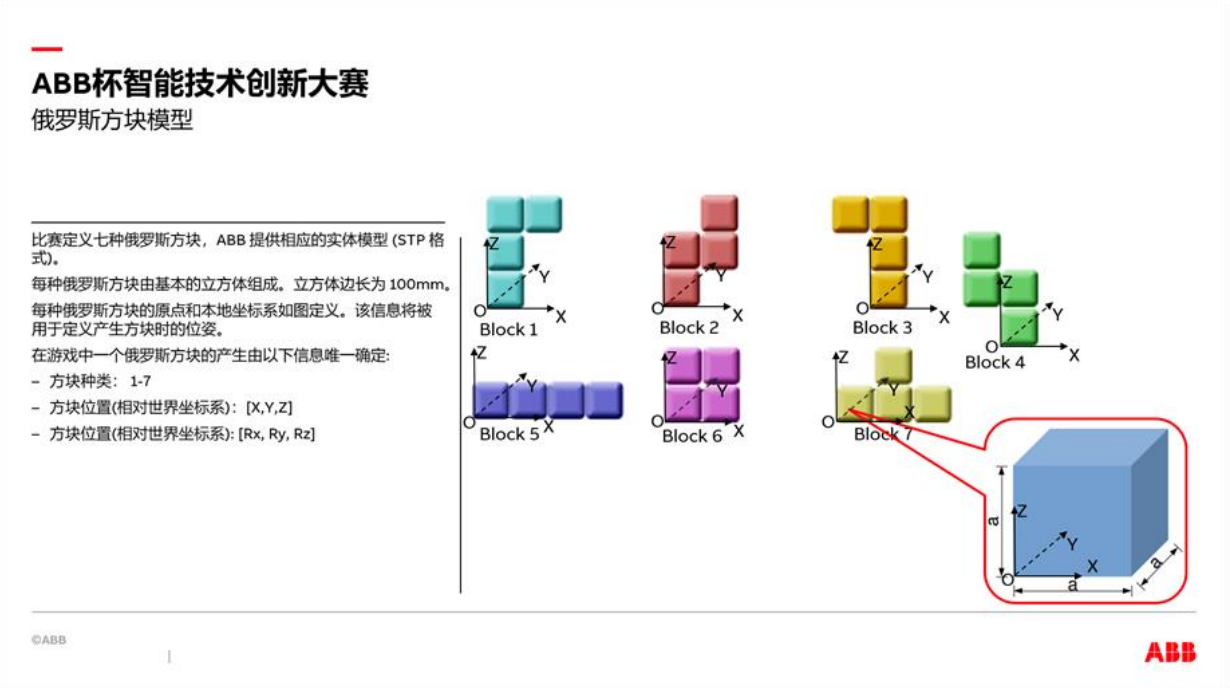


图1.2 俄罗斯方块模型

1.1.3 游戏规则

（1）设定游戏时间（10分钟），将俄罗斯方块放入仓库后，任一行满格后自动消除该行，高度自动往下降一行，并自动积分一分（参照传统俄罗斯方块的游戏规则）；

（2）满足以下任一条件，游戏结束：

a：给定的游戏时间达到（10分钟）

b：方块堆满仓库；

（3）以积分高且算法优的参赛者胜出。

1.2 赛题分析

1.2.1 工作区域划分和坐标系定义

如图1.1所示，本赛题将比赛区域区分为：方块产生区（L1）、机器人区（L2）和方块搭建区（L3）。其中，L1和L2区为1005*1005的长方形，L3区为1005*2000*105的矩形；

随机生成的俄罗斯方块的俯视图投影需位于L1区内，且不能生成在空中；机器人的俯视图投影需位于L2区内；放置后的俄罗斯方块需要在L3矩形体内；

用户坐标系{*U-XYZ*}：如图1.1、图1.3，位于L2区的中心位置，{*U-XYZ*}方向如图1.1所示，相对于世界坐标系{*O-XYZ*}的坐标为：[1507.5,502.5,0][0,0,0]（Position XYZ，mm；Orientation RxRyRz EulerZYX，deg；下文同），软件中名称USC；

世界坐标系{*O-XYZ*}：如图1.1、图1.3，位于L1区的左前方顶点位置，方向同{*U-XYZ*}，为软件默认的世界坐标系；

工件坐标系{*W-XYZ*}：每种俄罗斯方块工件坐标系方向及位置的定义如图

1.2、图1.3所示，相对于世界坐标系的位姿随机生成，软件中名称Workobject_1；

机器人坐标系{*R-XYZ*}：如图1.3所示，机器人相对世界坐标系{*O-XYZ*}的坐标为[1290, 560, 300][0,0,-90]，软件中名称wobj0（因软件设置习惯原因，图1.3中无名称显示）；

方块搭建区坐标系{*B-XYZ*}：如图1.3所示，方块搭建区（L3区）坐标系相对{*O-XYZ*}的坐标为：[2060,1000,0][90,0,-90]，软件中名称Workobject_2；

工具坐标系{*T-XYZ*}：如图1.3所示，具体见下文：2.1.2节机械臂末端执行器设计，及图2.2 (b)工具坐标系{*T-XYZ*}，软件中名称ToolFrame。

整体坐标系布置如图1.3所示：

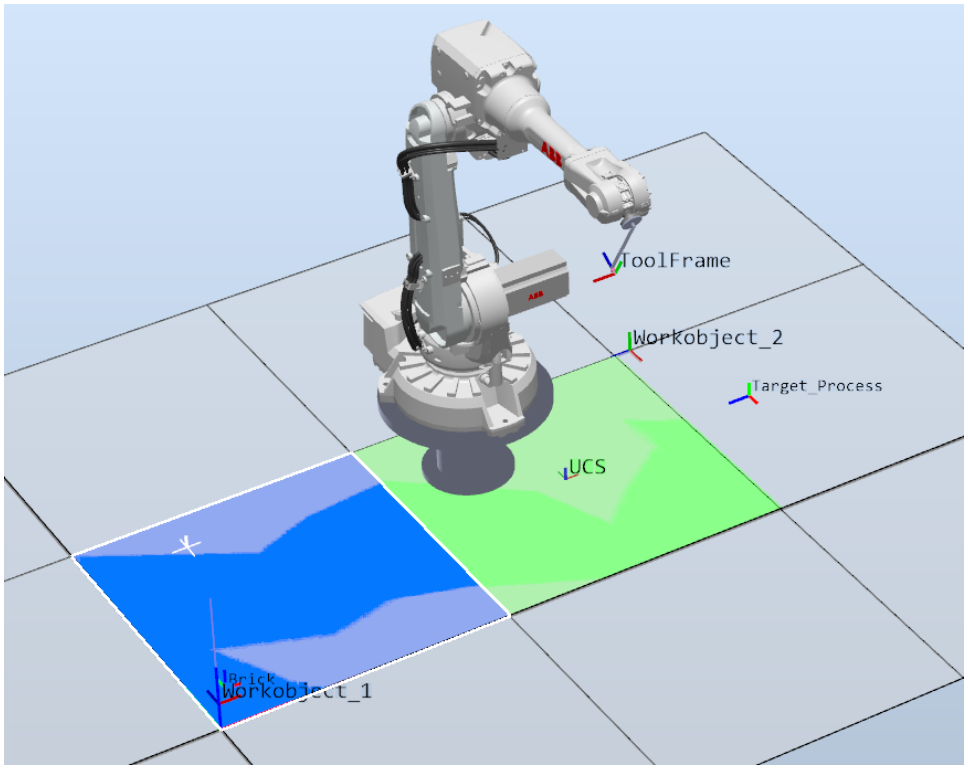


图1.3 坐标关系图

1.2.2 俄罗斯方块尺寸、位姿定义及随机生成范围

俄罗斯方块的小方块为100*100*100的矩形，俄罗斯方块本地坐标位于如图1.2所示，有7种类型，被7个参数完全定义：方块标识数（ID）、方块位置（x、y、

z）、方块姿态（Rx、Ry、Rz）；

按题干要求作者理解为：俄罗斯方块工件坐标{*W-XYZ*}相对于位于L2区域中心的世界坐标系{*O-XYZ*}变化的参数实质只有4个——方块标识数ID、横坐标x、纵坐标y、绕z轴的旋转角度Rz；其余数值均设为定值，即，Rx = Ry = 0 deg，z = 100 mm，随机的ID、x、y、Rz使用.net framework里的RNGCryptoServiceProvider来生成；俄罗斯方块使用如图一的姿态随机生成于L1区域，且方块边界不允许超出L1区域边界。

1.2.3 机器人型号、位姿及运动过程

机器人型号自选，机器人夹具自选或自制；机器人运动过程中不应该与其他物品发生干涉。

1.2.4 俄罗斯方块游戏规则

（1）7种类型俄罗斯方块在L1区域内随机生成，详细规则见——1.2.2节俄罗斯方块尺寸、位姿定义及随机生成范围；

（2）俄罗斯方块满行自动消除，并且上方的俄罗斯方块下落高度为消除的行数乘以俄罗斯小方块高度（本文中为单个小方块高度为100 mm）；

（3）俄罗斯方块移到区域最下方或是着地到其他方块上时其就会固定在该处；

（4）当游戏时间到达10min或者图形堆叠到游戏区域最顶端（本文中即达到20行）时，游戏结束；

（5）俄罗斯方块每消除一行积1分；

（6）每种俄罗斯方块能且只能以90度为单位旋转；

（7）机器人搬运俄罗斯方块的过程中不允许有碰撞发生；

（8）俄罗斯方块放置规则，有以下两种可能，各做了一种AI：

（8.1）经典俄罗斯方块规则，哪怕上方方块下有空位，只要上方有方块封闭路径，就不允许放置，对应的AI算法为：AI_No_Fill_In；

（8.2）进阶俄罗斯方块规则，俄罗斯方块下方有空位，允许机器人从侧面进行方块放置，对应的AI算法为：AI_Fill_In；

条件8.1和8.2只能同时存在一个，其余为通用规则。

2 机器人AI俄罗斯方块解决方案

2.1 机器人及夹具的选择安装

2.1.1 机器人选型与放置位姿

本方案主要依据所需工作空间对机器人进行选型并确定其在L2区内的放置位姿。

工作空间是用于衡量机器人末端执行器工作范围的重要参数，分为可达工作空间、灵巧工作空间和灵活工作空间；可达工作空间指：机器人可以到达的空间点的集合，可达工作空间求法主要有解析法、以蒙特卡洛法为代表的数值法和几何法，在RobotStudio软件内可以自动求解可达工作空间；灵巧工作空间指：机器人可以以任意姿态都可以到达的点的集合，是可达空间的子集，由于机械结构的限制，实际使用的机器人的灵巧工作空间是不存在的；灵活工作空间是机器人能以尽可能多的姿态到达的点的集合，是可达工作空间的子集，机器人灵活程度可以用可操作度等指标来描述，灵活工作空间的求解至今是学界在探讨的问题[1]，至今没有成熟的解决方案。

为了确保机器人能以期望位姿抓取或者放置随机生成的俄罗斯方块至指定区域，至少机器人安装末端执行器后的可达工作空间需要完全包络L1区（高度z=100，长方形面积为x*y=1005*1005）和L3区（长方体体积为x*y*z=1005*105*2000）。

经过测量与反复测试，最终选定机器人型号为：IRB2600_12/165，机器人坐标系{*R-XYZ*}相对于世界坐标系{*O-XYZ*}的位置和姿态为：[1280,730,300][0,0,-90]（rapid程序内默认使用机器人坐标系{*R-XYZ*}作为基坐标）。最终的场景布置如图2.1所示。

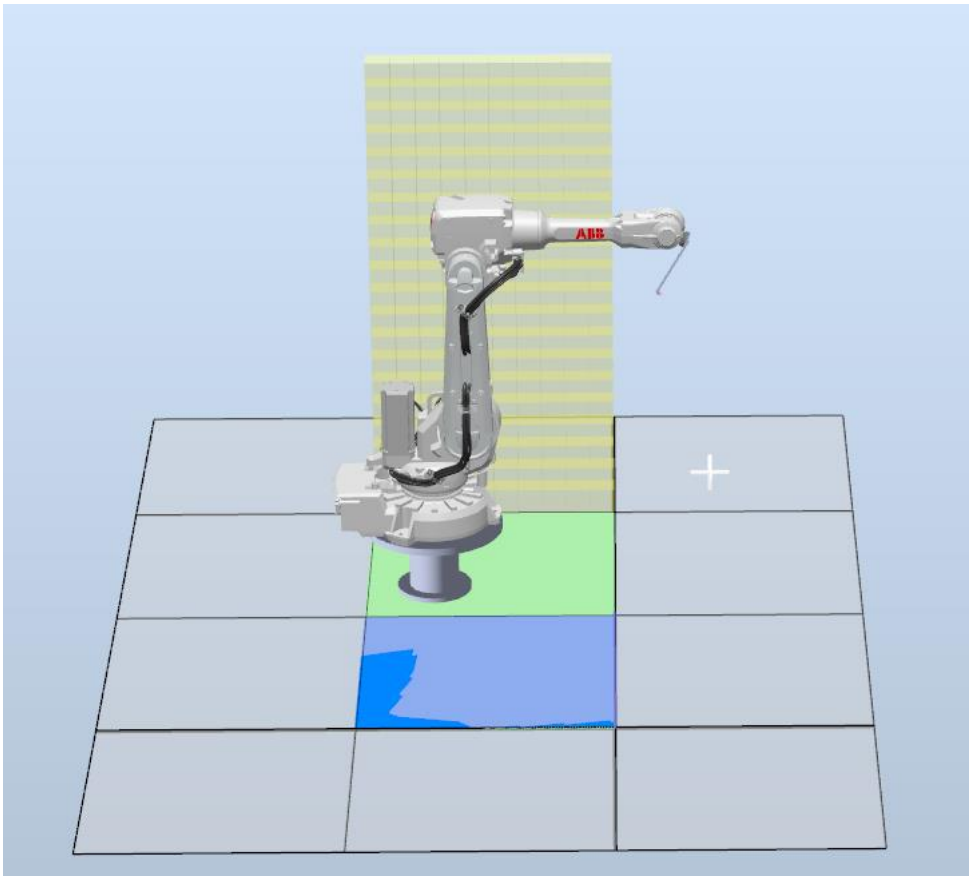
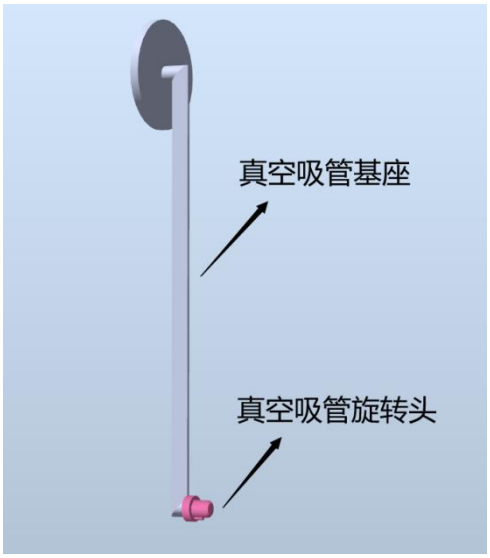
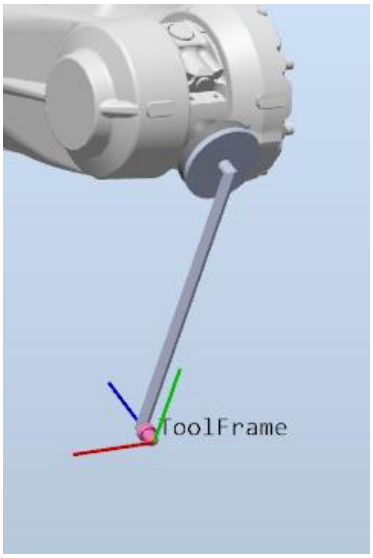


图2.1 仿真场景布置

2.1.2 机械臂末端执行器设计



(a) 气动吸盘组成



(b) 工具坐标系{*T-XYZ*}

图2.2 机器人单自由度末端气动吸盘

为了增加机器人的灵活度，本方案设计了如图2.1所示的单自由度气动吸盘，

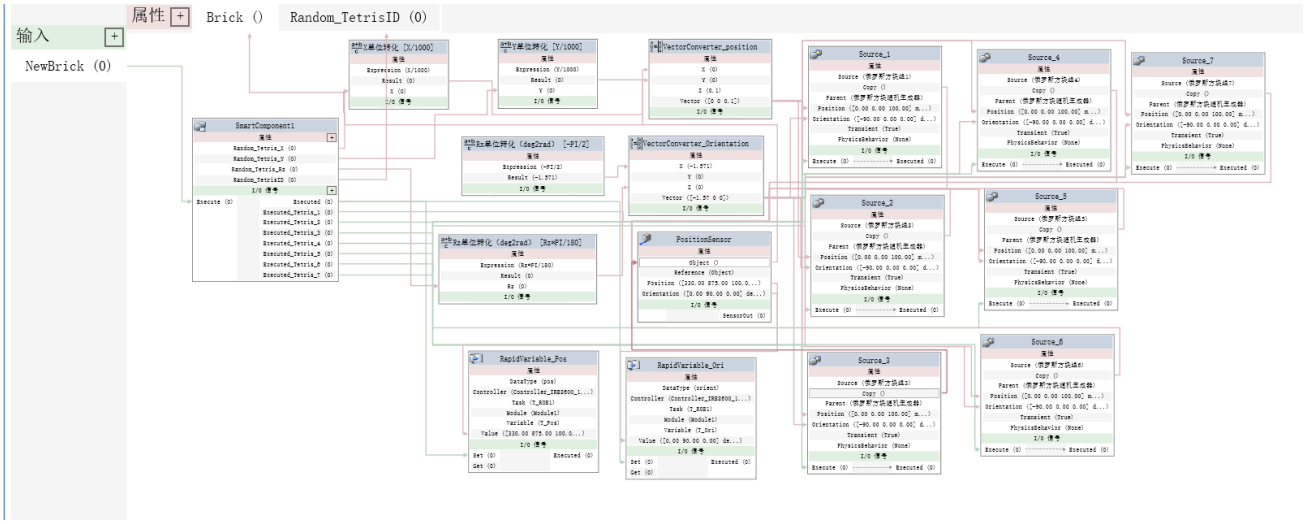
详见附件3——机器人单自由度末端吸盘My_Mechanism。该气动吸盘的真空吸管旋转头有绕着其中心轴线的单自由度（DoF, Degree of Freedom）旋转，安装上该末端执行器后机器人实际为7-DoF机械臂，安装于机器人IRB2600_12/165的第六轴，其工具坐标系如图2.2 (b)所示。

当机器人吸取俄罗斯方块后，该气动吸盘的真空吸管旋转头会旋转方块至下文2.5小节AI算法计算出的位置。

2.2 Smart组件

2.2.1 基于RobotStudio SDK的随机生成俄罗斯方块Smart组件

(a) 随机数集成模块



(b) 随机数集成模块连接

图2.3 随机数集成Smart组件

本方案使用VisualStudio软件，基于RobotStudio SDK（以下简称为RS SDK）制作随机生成俄罗斯方块的Smart模块，使用.net framework里的RNGCryptoServiceProvider来生成随机数，该模块用于生成四个随机数——俄罗斯方块标识数ID、世界坐标系下的俄罗斯方块横坐标x、世界坐标系下的俄罗斯方块纵坐标y、俄罗斯方块绕自身坐标系y轴（世界坐标系下的z轴方向）的旋转角度Rz（角度制）；考虑到这些数值的实际意义，如俄罗斯方块只有7种类型，所以设置$ID \in [1,7], ID \in Z^+$，考虑到俄罗斯方块在任何情况下都不能出界，设置如图二所示的俄罗斯方块类型1、2、3、4、7的x、y范围为$x、y \in [300,700]$ mm，罗斯方块类型5的x、y范围为$x、y \in [400,600]$ mm，俄罗斯方块类型6的x、y范围为[200,800] mm；其C#代码文件见附件1——随机生成俄罗斯方块Smart模块，该模块如图5 (a)所示，需要在RobotStudio（以下简称为RS）内建立整个随机生成俄罗斯方块Smart的组件，其连接方式如图2.3(b)所示。

其有4个属性：

Random_Tetris_X (Double)：在世界坐标系{*O-XYZ*}下俄罗斯方块的*X*方向随机值，mm；

Random_Tetris_Y (Double：在世界坐标系{*O-XYZ*}下俄罗斯方块的*Y*方向随机值，mm；

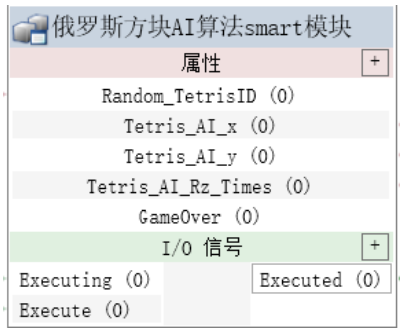
Random_Tetris_Rz (Double) ：在俄罗斯方块绕自身*Y*轴（工件坐标系{*W-XYZ* }的*Y*轴，与世界坐标系{*O-XYZ*}的Z轴同向，因此命名为Rz）方向旋转随机值，deg；

Random_TetrisID (Double) ：俄罗斯方块类型识别数，如图1.2所示，为1到7的正整数。

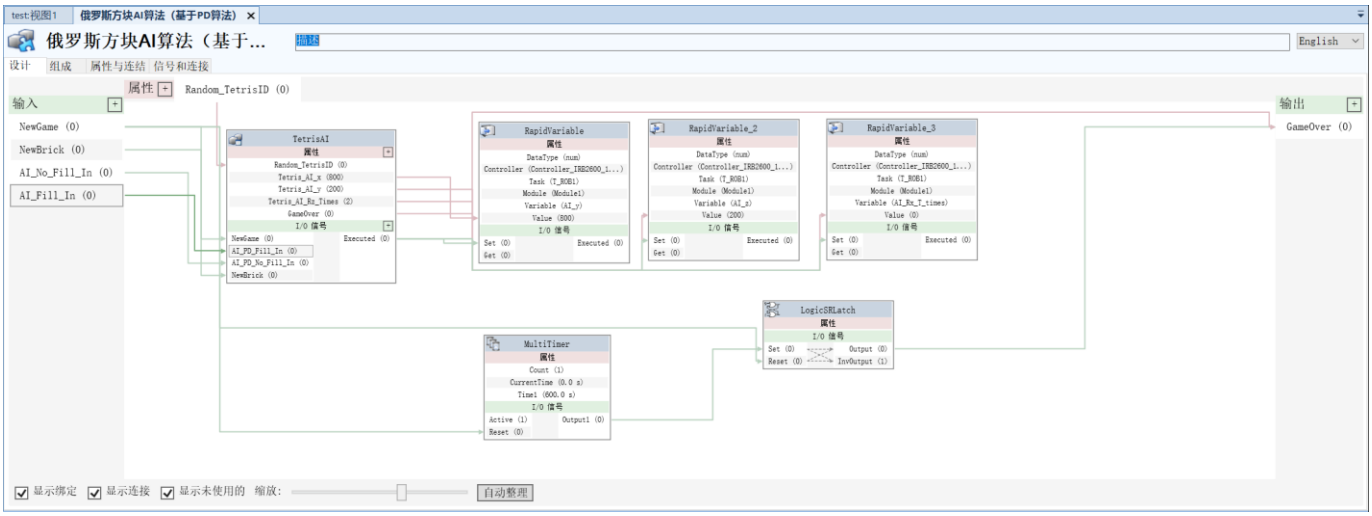
使用Expression进行单位转化，将mm转化为m，将角度制转化为弧度制。使

用Source复制产生各种不同的俄罗斯方块，由于其相对世界坐标系{*O-XYZ*}生成，因此需要进行坐标转化，*z*坐标固定为0.1m，绕自身工件坐标系{*W-XYZ*}*x*轴旋转始终为90度，其余值使用随机数集成模块随机生成，其RS SDK的C#代码见附件1——随机生成俄罗斯方块Smart模块源代码。

2.2.2 基于RobotStudio SDK的俄罗斯方块AI算法Smart组件



(a) 俄罗斯方块AI算法Smart模块



(b) 俄罗斯方块AI算法Smart模块连接

图2.4 俄罗斯方块AI算法Smart组件

俄罗斯方块AI算法Smart模块（图2.4(a)）用于根据当前Random_TetrisID生成俄罗斯方块在世界坐标系{*O-XYZ*}下的放置位置Tetris_AI_x、Tetris_AI_y，和俄罗斯方块绕工具坐标系{*T-XYZ*}的*Z*轴的旋转次数Tetris_AI_Rz_Times三个参数，并判别是否俄罗斯方块上越界而导致游戏结束，使用bool型参数GameOver表示，

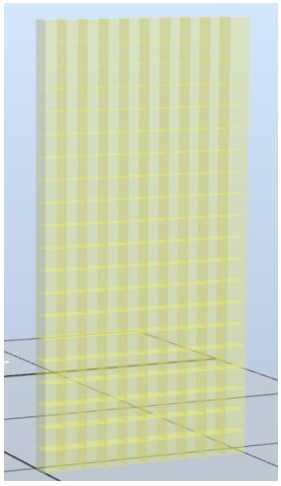
置1则游戏结束；其中有两种AI，都是基于PD算法，区别是是否运行机器人从侧边插空进行俄罗斯方块游戏，AI_No_Fill_In表示不允许插空，为经典的俄罗斯方块游戏规则下的AI，即哪怕下方有空位，只要上方封闭就不允许放置方块到该位置；AI_Fill_In表示允许插空，即允许机器人从侧面用方块填补空缺的位置。

建立如图2.4 (b)所示的Smart组件，使用模块：RapidVariable（3个）将俄罗斯方块放置位姿的3个参数Tetris_AI_x（double型）、Tetris_AI_y（double型）、Tetris_AI_Rz_Times（int型）传递至Rapid程序。通过上述Smart组件，当时间到达10 min或者俄罗斯方块越界时程序停止。其核心Smart模块——俄罗斯方块AI算法Smart模块的建立思想及代码见下文，2.4节俄罗斯方块自动消除的AI算法。

2.2.3 用于消行下落的Smart组件

为了模拟俄罗斯方块满行消除以及消除后被消除行上方俄罗斯方块整行下落的动作，需要利用到ABB Studio 软件中Smart组件来组合实现，具体实现方法及思路如下：

(1) 消行下落的Smart组件建立及传感器的设置



(a) MatrixRepeater Smart模块　　(b) 10*20的俄罗斯方块矩阵

图2.5 10*20的俄罗斯方块区域生成

首先根据比赛要求需建立一个20*10的俄罗斯方块游戏界面，俄罗斯的方块

的模型为100mm*100mm*100mm的正方体方块，因此在新建工作站之后，开始方块模型的建立，利用创建方块功能，在世界坐标的原点位置创建一个符合标准的正方体模型，再利用图2.5 (a)所示的MatrixRepeater Smart组件对正方体方块对源正方体方块进行阵列，得到图2.5 (b)所示的10*20的俄罗斯方块矩阵。

为了配合线性传感器的扫描动作需要建立一个装置来移动传感器，方便扫描到所有方块，使用创建实体功能，创建一个长为1000mm，宽为100mm，高为10mm的长杆模型杆L_1，创建完毕以后进行复制，再绕末端顶点旋转90°得到杆L_2，结果如图2.6所示。

首先将杆L_1设置机械装置，将杆L_2放置入杆L_1之中，并将L_2设置为往复关节，位移的方向为*x*的负方向，移动范围为[0-1000]mm，每次移动的step设置为100mm，将创建的移动杆移动到游戏界面矩阵的对应位置。

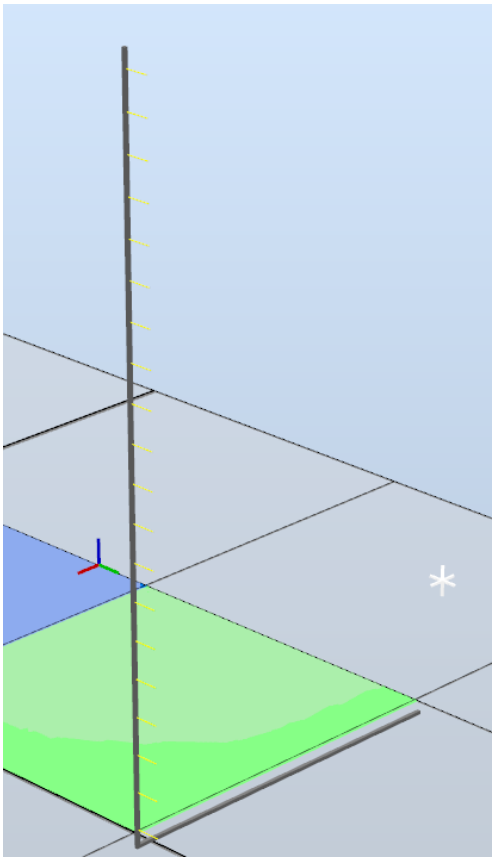


图2.6 移动机械装置设置

(2) Smart组件的设置与连接

新建一个Smart组件Joint Move与当前关节相关联，移动步长与时间与移动杆一致，当Execute信号激发时，会在前一次移动step的基础上进行下一步的移动。新建一个Repeater脉冲组件，次数count设置为10，设置一个Pose Move组件来使每次扫描完成之后，移动杆的移动关节回到初始位置，Pose Move的激发时间设置为24 ms,因为该组件在脉冲信号完成后来触发，所以需要设置一个Logic gate组件来对信号进行转换。

创建一个用于扫描的Smart组件，并在里面新建一个Line Sensor的直线传感器，具体姿态如图2.6所示。

移动杆每个step移动完成之后，就给扫描组件传递一个Executed信号，同时给扫描组件一个输入信号，在该Smart组中再创建一个queue组件，每次传感器感应到一个方块之后就会传递一个信号，然后将对应方块加入到阵列之中，并显示阵列中方块的个数，当一个阵列中满方块时，Number of Object会显示10。同时新建一个Compare组件，用以判断queue中方块的个数是否等于10，即判断是否满行，当检测到满行时，通过一个Logical Gate 组件传递信号给queue组件删除其中所有方块，来完成消行。

其中为了避免在上一次检测的时候，队列中的元素未满，无法满足删行的条件使得方块数依旧保存在Queue阵列中，因此在Compare组件出发的0.1s之后，会自动产生一个信号激活Queue的Clear功能清除剩余的方块信息，同时Hide组件的存在是为了及时将满行的阵列隐藏，并等待删除。

下一步需要模拟俄罗斯方块满行下落的动作，由于满行消除是在瞬时以及同时完成的，因此在上方的阵列需要下落的行数可能大于1，因此需要设计一个信

号传递来进行计算。通过新建一个输入属性的接口Detection Signal，以及输出属性接口Output。新建一个Express组件，组件的变量设为a、b，Result结果为0,Expression公式为a+b，将上一行计算的Result，下一行将上一行传递的Result视为b，当Compare组件确认满行，b的数值就会被自动赋1否则赋0，以此来叠加计算每一行所需的下落行数，如图2.7满行检测与下坠组件所示。

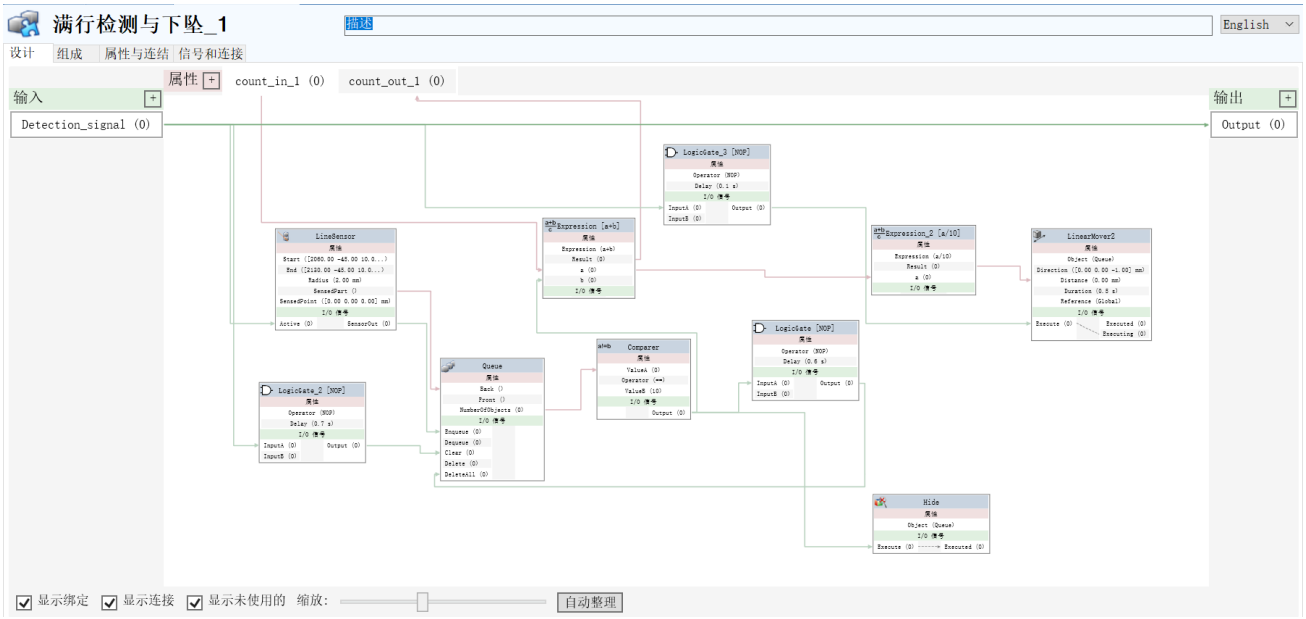


图2.7 满行检测与下坠组件

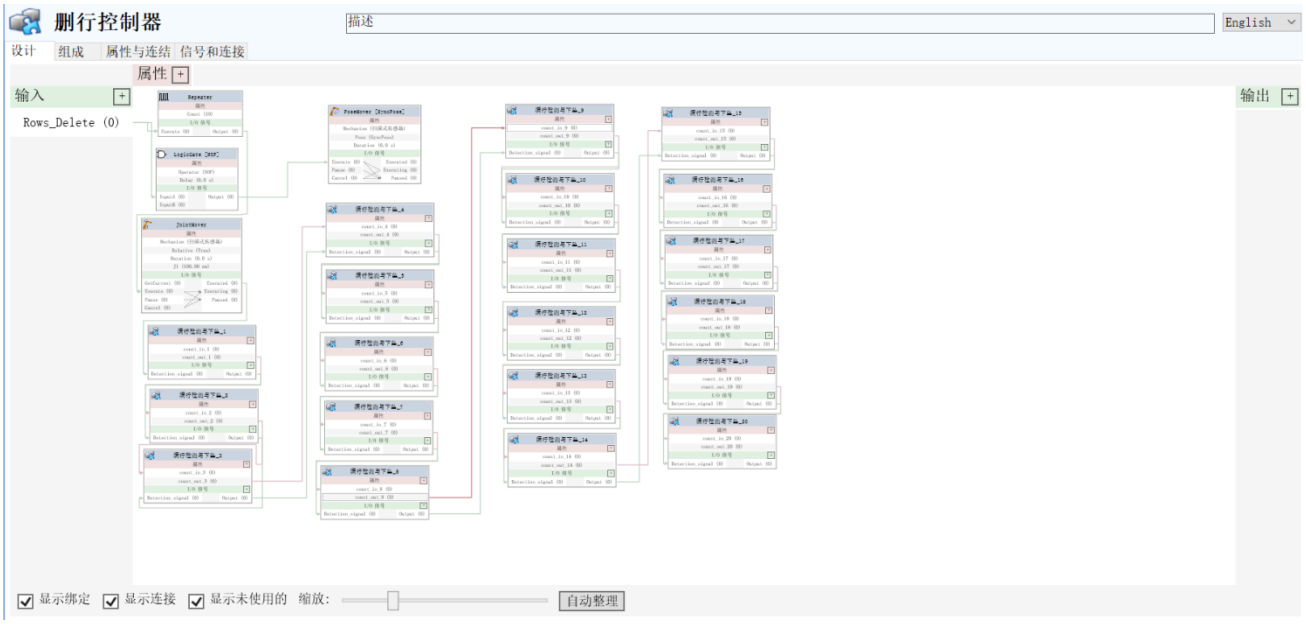


图2.8 删行控制器组件

在完成下落行数的计算之后，要开始进行下落动作的模拟，新建一个

LinearMover组件来将阵列移动到一个指定位置，其中Expression2组件用以将上一行累计的消行数a的值传递到LinearMover的Distance中去，移动时间Duration设置为1s，行下落的动作必须要在线性传感器归位动作开始之前完成。

由于所有阵列完成需要20个传感器，所以下一步需要对扫描组件进行复制操作，将各个扫描组件的输入和输出进行连接来进行信号的传递，整体连接图如图2.8所示。

2.2.4 机器人吸盘的Smart组件

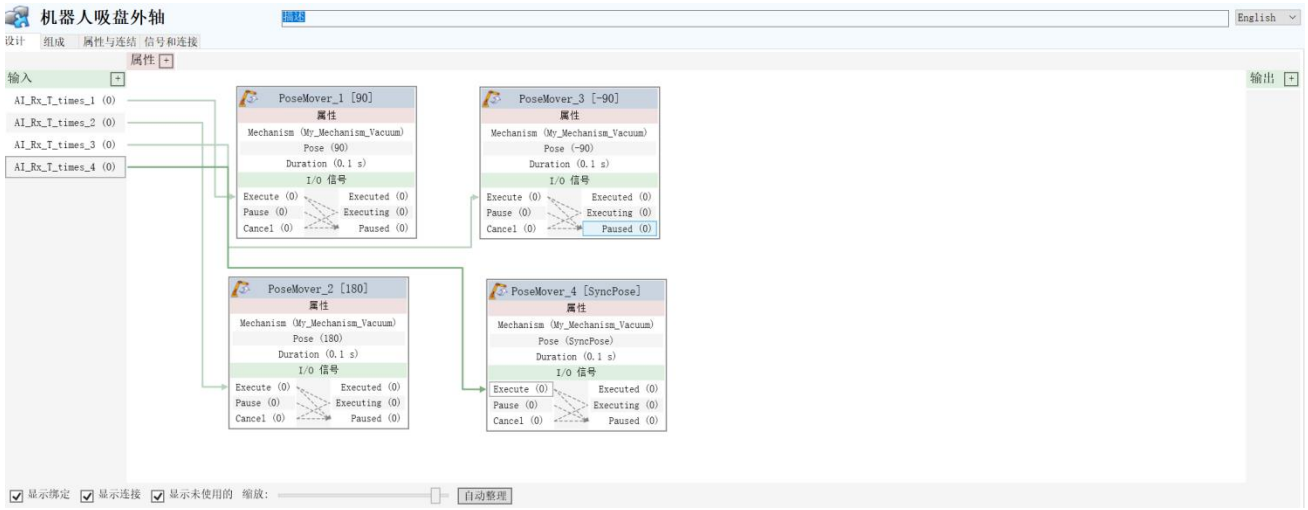


图2.9 机器人吸盘外轴

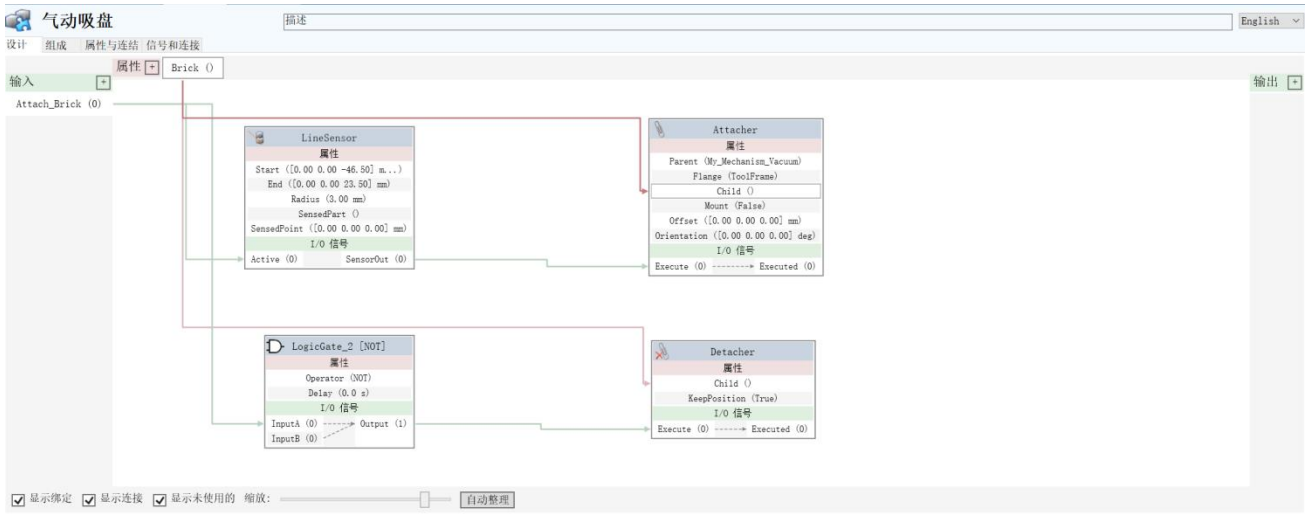


图2.10 气动吸盘

建立如图2.9所示的机器人吸盘外轴Smart组件，输入为AI_T_Times_1（对应

90°）、AI_T_Times_2（对应180°）、AI_T_Times_3（对应-90°）、AI_T_Times_4（对应原点0°），当这四个信号其一给到对应的PoseMover时，机器人吸盘外轴将会逆时针转动到90°、180°、-90°或原点，时间设置为0.1s。

建立如图2.10所示的气动吸盘Smart组件，当输入Attach_Brick有信号时，给安装于气动吸盘末端的线传感器LineSensor激活信号，当LineSensor检测到物体时，LineSensor的SensorOut给信号至Attacher，新生成的俄罗斯方块被安装于气动吸盘，当Attach_Brick无信号时，经过非门LogicGate的信号输出Output给出信号至Detacher，物品被从吸盘上卸下。

最终所有的组件列表如图2.11所示，其中 图标表示未隐藏的Smart组件， 图标表示隐藏的Smart组件， IRB2600_12_165_C_01_2 表示机器人及其型号， My_Mechanism_Vacuum 为自制工具， UCS 为隐藏的用户坐标系， 表示几何体。

图2.11 组件列表

2.3 组件间通讯

组件间通讯关系如图2.12所示，控制柜Controller有输入信号GameOver0控制其结束程序，有输出信号NewGame0、NewBrick0、Attach_Block0、Rows_Delete0、AI_Rx_T_times_10、AI_Rx_T_times_20、AI_Rx_T_times_30、AI_Rx_T_times_40、AI_No_Fill_In、AI_Fill_In，分别控制开新局、产生新方块、抓取/放下方块、删行与检测、旋转1（90°）、2（180°）、3（-90°）、0次（0°）、选择不插空版AI、选择插空版AI。

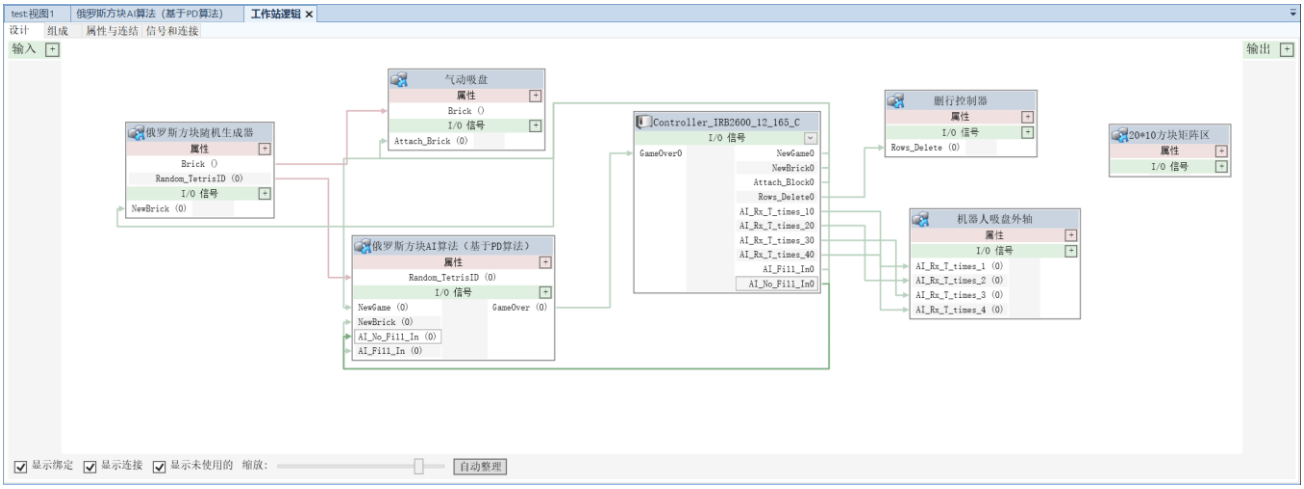


图2.12 工作站逻辑(详见工作站)

2.4 机器人Rapid程序思路及轨迹规划

为了在RobotStudio中实现机器人在方块产生区L1抓取随机生成的俄罗斯方块，并堆放俄罗斯方块至方块搭建区L3。Rapid程序中包含了一系列的控制机器人的指令，这些指令使机器人能够实现所需的操作。

具体Rapid程序实现及思路如下：

定义变量俄罗斯方块在初始状态下方块标识数ID，俄罗斯方块在世界坐标系{*O-XYZ*}下的放置位置AI_y、AI_z，和俄罗斯方块绕工具坐标系{*T-XYZ*}的*Z*轴(即世界坐标系{*O-XYZ*}的*X*轴反方向)的旋转次数AI_Rx_T_Times，这四个参量初值

均为0。定义变量目标点坐标T_Pose，该目标点包含目标点位置T_Pos和目标点姿态T_Ori这两部分，用于确定随机产生的工件坐标系{*W-XYZ*}（Workobject_1）坐标，定义W1_Pos作为中间变量。定义俄罗斯方块搭建区坐标系{*B-XYZ*}（Workobject_2）的位置W2_Pos，和相对{*B-XYZ*}的AI_Pos。

在工作站中，在工件坐标系{*W-XYZ*} (Workobject_1)下建立工件目标点Brick，用于确定机器人抓取俄罗斯方块时的位姿。机器人坐标系{*R-XYZ*}（wobj0）下建立机械臂中间途径点Target_Process，用于避免碰撞；在俄罗斯方块搭建区坐标系{*B-XYZ*} (Workobject_2)下建立工件目标点Target_AI，用于确定俄罗斯方块被放置的位姿；{*W-XYZ*}、{*B-XYZ*}和{*T-XYZ*}被一起同步到Rapid程序。

接着编写Rapid程序的主函数。首先将机器人以7000m/s的速度沿非线性路径运动至零点（绝对位置点[[0,0,0,0,30,0]），先设置信号NewGame0的数字信号为0即控制新局开始信号为零，再将NewGame0的数字信号设置为1即新局开始。再选择AI类别，插空版AI_No_Fill_In0或者不插空版AI_Fill_In0。

当GameOver0不等于1时，执行以下程序：初始化坐标{*W-XYZ*}(Workobject_1)、{*B-XYZ*}(Workobject_2)。然后输出信号NewBrick0产生长度为0.1s的方波信号，令RobotStudio开始产生新方块，程序执行后等待0.1s，将目标位置坐标T_Pos和T_Ori形成的数组存储到T_Pose中，更新后的工件坐标系Workobject_1.uframe为数组T_Pose的数据点。将更新后的工件坐标位置W2_Pos加上[AI_y,0,AI_z]的位置坐标得到更新后的俄罗斯方块搭建区的位置坐标AI_Pos。更新后的位置坐标AI_Pos组成了工件坐标系Workobject_2的移动数组。在工件坐标系{*W-XYZ*}(Workobject_1)下机械臂沿非线性路径移动精确到达Brick点，其速度为7000m/s。到达位置后，设置信号Attach_Block0的数字信号为1，吸

盘抓取生成的方块。

接着判断生成的方块是否离机器人太近，太近则先将方块移动到即使是最差的情况也不会发生碰撞的位置（四个小方块一条直线的情况加上机器人底座半径的距离）；再将机械臂抬起200mm，接着让机器人走到中间点位Target_Process点，避免碰撞和可能产生的绕路等不期望的位姿。

接着对移动到该位置的方块进行判断，如果AI_Rx_T_times等于1则输出信号AI_Rx_T_times_10产生脉冲长度为0.2s的脉冲即抓取的俄罗斯方块旋转1(90°)次，如果AI_Rx_T_times等于2则输出信号AI_Rx_T_times_20产生脉冲长度为0.2s的脉冲即抓取的俄罗斯方块旋转2(180°)次，如果AI_Rx_T_times等于3则输出信号AI_Rx_T_times_30产生脉冲长度为0.2s的脉冲抓取的俄罗斯方块旋转3(-90°)次，如果AI_Rx_T_times等于4则输出信号AI_Rx_T_times_40产生脉冲长度为0.2s的脉冲抓取的俄罗斯方块旋转0(0°)次，判断AI_Rx_T_times的值的四种情况则跳出if循环。

为了避免碰撞，机械臂先将俄罗斯方块放置于堆放区L3往机器人方向偏置100mm的预堆放区，调整好位姿，再将俄罗斯方块塞进去。

设置信号Attach_Block0的数字信号为0即放下方块，输出信号Rows_Delete0产生脉信号进行删行与检测，输出信号AI_Rx_T_times_40产生脉信号令气动夹爪转头位置回到原点，此时进行while循环判断，若不符合while循环，则程序结束。

子函数Path_1并不会执行，只用于同步工作站与Rapid程序的坐标、工具等数据。Rapid程序见附件5——solution1解包后的工作站控制器内程序。注意，Rapid程序运行前请热重启控制器，以免运行过程中出现意外的动作/信号，仿真中的保存状态Reset为——演示界面。

2.5 俄罗斯方块自动消除的AI算法（2种）

在1997年Heidi Burgiel根据极大极小值算法证明了完全随机的俄罗斯方块游戏最终一定会结束。因此俄罗斯方块的AI算法专注于怎样才能让俄罗斯方块游戏得到更高的分数。俄罗斯方块AI算法主要可以分成只考虑当前块的one-piece算法和考虑当前块和下一块的two-piece两种，根据实践证明one-piece算法并不劣于two-piece算法，且算法更加简单，因此本方案使用one-piece算法中的经典算法——Pierre Dellacherie算法（以下简称为PD算法），该算法被誉为最好的俄罗斯方块one-piece算法之一。

PD算法摆放一个板块的策略是：板块放置的位置越靠下越好，方块之间越紧密越好，自身对消除行的方块贡献数量越多越好。但不可为了追求消除行数，而去造成过多的空洞，这样也是不合理的。

因此该算法有以下6个具体参数用于评价摆放位置的优劣（含义一致，单词使用可能因个人习惯而异）：

landingHeight：指当前板块放置之后，当前板块重心距离游戏区域底部的距离。（也就是小方块的海拔高度）；

rowsEliminated：指消行层数（clearRows）与当前方块贡献出的用于此次消行的方格数（contribution）的乘积；

rowTransitions：对于每一行小方格，从左往右看，从无小方格到有小方格是一种“变换”，从有小方格到无小方格也是一种“变换”，每次变换计为1，这个属性是各行中“变换”之和；

columnTransitions：这是每一列的变换次数之和，从上往下看；

holes：各列中上方有方块的“空洞的小方格数之和”；

wellSum：各列中“井”的深度的连加和（从当前深度累加到1为止），“井”的定义是，两边（包括边界）都有方块填充的空列，有些空洞既是洞又是“井”。

计算公式为：

result=index1*landingHeight+index2*clearRows*contribution+index3*rowTransitions+index4*columnTransitions+index5*holes+index6*wellSum；

式中，index1~index6为经验值，由粒子群算法计算得出，其具体数值见下表2.1：

表2.1 俄罗斯方块AI算法之PD算法系数经验值

Index	Value
1	-4.500158825082766
2	3.4181268101392694
3	-3.2178882868487753
4	-9.348695305445199
5	-7.899265427351652
6	-3.3855972247263626

result值越大越好，设立碰撞检测方法CanMove用于检测测试方块是否能移动到目标位置（俄罗斯方块不能出界或者和已有块重叠），在当前方块下落之前，虚拟测试俄罗斯方块（testBrick）会从左到右、从下到上、从逆时针旋转0次到3次，遍历所有本次俄罗斯方块放置位置的可能性，取result值最大的结果作为实际下落位置，如果result值相同取最初的结果。

相当于在C#程序内建立了一个虚拟环境计算俄罗斯方块的位姿，然后映射至RS程序内，建立连接关系。其数据传递方式见——2.2.2节基于RobotStudio SDK的俄罗斯方块自动消除的AI算法Smart组件，2.3节组件间通讯。

插空与不插空版的区别在于循环读取可放置位置时，是从第20行往第1行读取当前列的高度，遇到有方块的位置记录当前列的高度（不插空版）；还是从第1行往第20行读取当前列的高度，遇到无方块的位置记录当前列的高度（插空版）；

其中利用List泛型列表制作板块，并撰写了碰撞检测方法。

具体代码见附件2——俄罗斯方块AI算法smart模块。

3 结果演示

如图3.1所示，仿真运行时在RS控制台（Output Message）会输出日志，解释当前使用的AI类型，俄罗斯方块ID（1-7），旋转标识数（1：逆时针旋转90°，2：逆时针逆时针旋转180°，3：顺时针旋转90°，4：不旋转），和放置点俄罗斯方块搭建区坐标系{*B-XYZ*}（Workobject_2）的下的x、y坐标；并且记录当前分数，消除1行记1分。

图3.2为机器人自主抓取随机生成的俄罗斯方块，并根据算法搬运俄罗斯方块至指定位置及俄罗斯方块自动消行的仿真运行过程中的截图。

执行基于PD算法的俄罗斯方块AI程序smart组件，每消除1行为1分
执行插空版AI
随机生成俄罗斯方块
俄罗斯方块ID为：3；旋转标识数：3；x坐标：700mm；y坐标：0mm
当前分数：0分

图 3.1 仿真时控制台输出

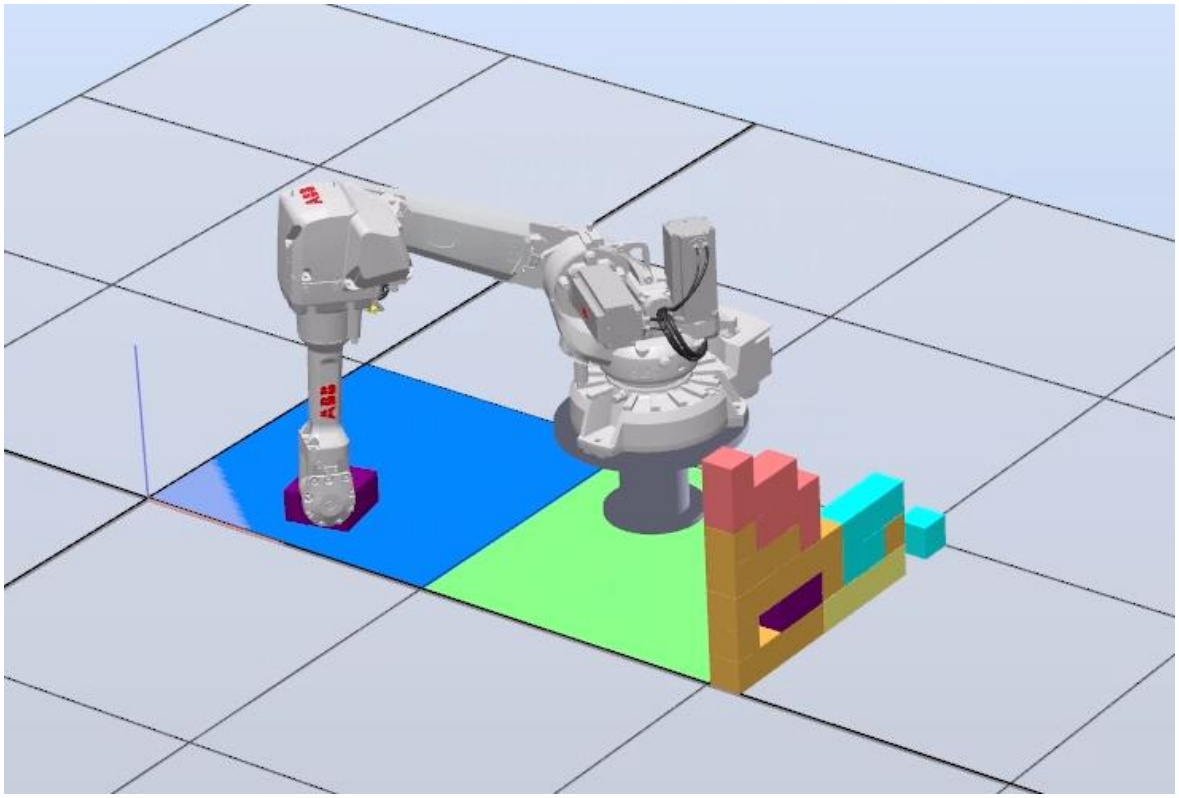
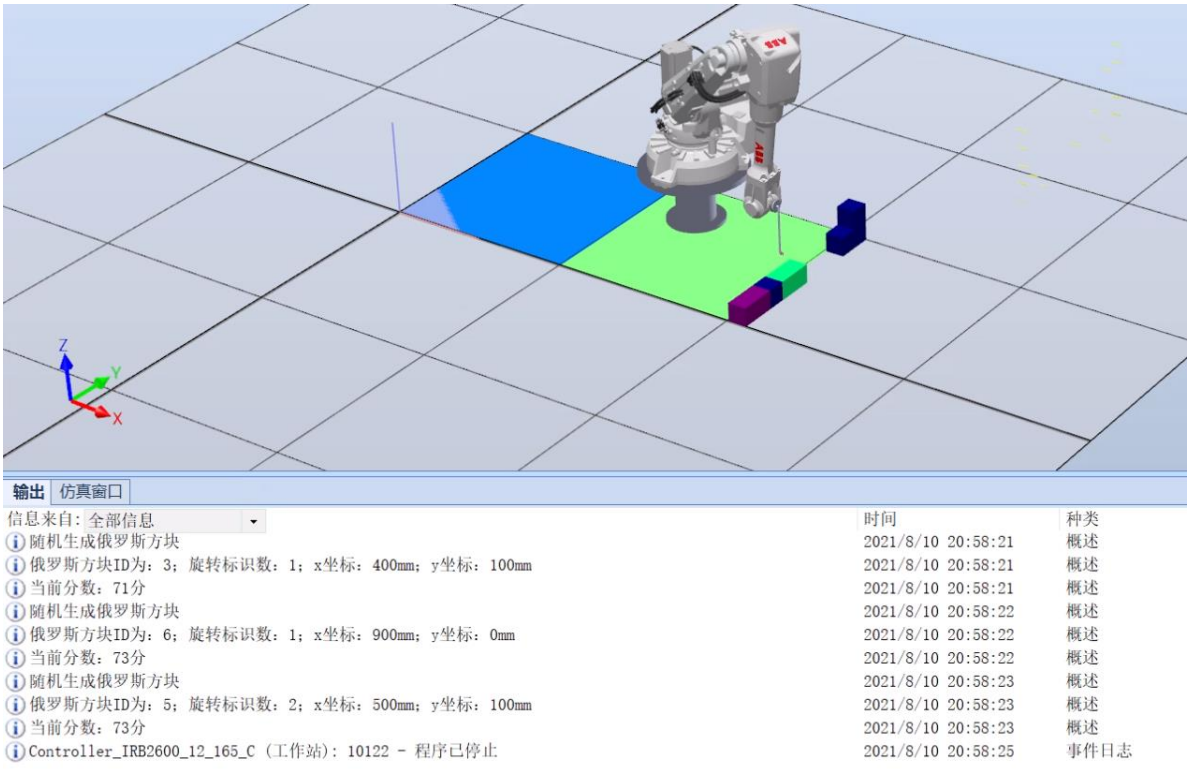


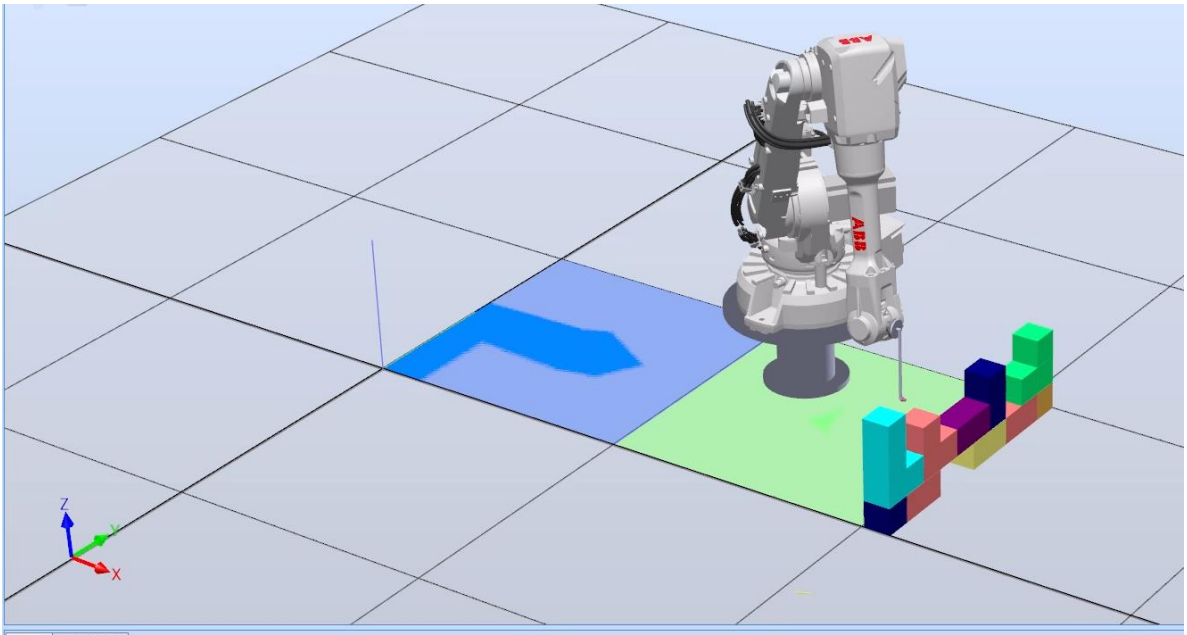
图 3.2 仿真运行中界面



输出 | 仿真窗口

信息来自：全部信息	时间	种类
随机生成俄罗斯方块	2021/8/10 20:58:21	概述
俄罗斯方块ID为：3；旋转标识数：1；x坐标：400mm；y坐标：100mm	2021/8/10 20:58:21	概述
当前分数：71分	2021/8/10 20:58:21	概述
随机生成俄罗斯方块	2021/8/10 20:58:22	概述
俄罗斯方块ID为：6；旋转标识数：1；x坐标：900mm；y坐标：0mm	2021/8/10 20:58:22	概述
当前分数：73分	2021/8/10 20:58:22	概述
随机生成俄罗斯方块	2021/8/10 20:58:23	概述
俄罗斯方块ID为：5；旋转标识数：2；x坐标：500mm；y坐标：100mm	2021/8/10 20:58:23	概述
当前分数：73分	2021/8/10 20:58:23	概述
Controller_IRB2600_12_165_C（工作站）：10122 - 程序已停止	2021/8/10 20:58:25	事件日志

(a) 插空版 AI



输出 | 仿真窗口

信息来自：全部信息	时间	种类
随机生成俄罗斯方块	2021/8/10 20:38:37	概述
俄罗斯方块ID为：4；旋转标识数：2；x坐标：0mm；y坐标：300mm	2021/8/10 20:38:37	概述
当前分数：72分	2021/8/10 20:38:37	概述
随机生成俄罗斯方块	2021/8/10 20:38:38	概述
俄罗斯方块ID为：1；旋转标识数：2；x坐标：900mm；y坐标：400mm	2021/8/10 20:38:38	概述
当前分数：72分	2021/8/10 20:38:38	概述
随机生成俄罗斯方块	2021/8/10 20:38:39	概述
俄罗斯方块ID为：2；旋转标识数：2；x坐标：700mm；y坐标：300mm	2021/8/10 20:38:39	概述
当前分数：72分	2021/8/10 20:38:39	概述
Controller_IRB2600_12_165_C（工作站）：10122 - 程序已停止	2021/8/10 20:38:40	事件日志

(b) 不插空版 AI

图 3.3 仿真结束时界面

如图3.3所示，本次仿真10min内2种AI下的俄罗斯方块均没有满20行，时间到达10min后游戏结束；插空版AI使用机械臂共消去了73行俄罗斯方块；不插空版AI使用机械臂共消去了72行俄罗斯方块。

本次具体运行情况见附件4（视频）与附件6（工作站文件）。

工作站见附件5。

4 队伍成员、分工与运行/开发环境

4.1 队伍成员

4.2 分工

4.3 使用软件、运行及开发环境

运行及开发环境：Windows 10（x64）

使用的软件/包：

1. RobotStudio 2020.4
2. VisualStudio 2019 Community（C#，.NET Framework 4.8）
3. RobotStudio SDK 2020.4
4. IRC5的RobotWare 6.11.03
5. SolidWorks 2018

5 参考文献

[1] 赵智远,徐振邦,何俊培,贺帅,徐策.基于工作空间分析的9自由度超冗余串联机械臂构型优化[J].机械工程学报,2019,55(21):51-63.

[2] Robotics操作手册:RobotStudio.ABB,2021,文档编号:3HAC032104-010,修订:AG.

[3] 技术参考手册RAPID语言概览:RobotWare 6.02.ABB,文档编号:3HAC050947-010,修订:B.

[4] 技术参考手册RAPID指令、函数和数据类型:RobotWare 6.02.ABB,文档编号:3HAC050917-010,修订:B.

[5] 技术参考手册RAPID语言内核.ABB,文档编号: 3HAC050946-010,修订: A

[6] ielashi.El-Tetris–An Improvement on Pierre Dellacherie's Algorithm开源项目. 官网地址:https://imake.ninja/el-tetris-an-improvement-on-pierre-dellacheries-algorithm/,代码地址:https://github.com/ielashi/eltetris (Java).2011.

[7] tetris_ai开源项目.代码地址:https://github.com/grdaimap/tetris_ai (C#).2020.

6 附件

附件1——随机生成俄罗斯方块Smart模块源代码及其生成的RobotStudio库文件：SmartComponent1.rslib；

附件2——俄罗斯方块AI算法Smart模块源代码及其生成的RobotStudio库文件：俄TetrisAI.rslib；

附件3——机器人单自由度末端吸盘库文件：My_Mechanism_Vacuum.rslib及SolidWorks源文件；

附件4——solution_AI_Fill_In.mp4，solution_AI_No_Fill_In.mp4（由于文档大小限制50M，视频经过压缩，视频仿真时速度为尽快）；

附件5——solution_2_AI.rspag（Pack&GO打包文件，内部组件均已断开库连接，每次Rapid程序运行前请热重启控制器，以免运行过程中出现意外的动作/信号，仿真中的保存状态Reset为——演示界面）；

附件6——solution_AI_Fill_In.exe，solution_AI_No_Fill_In.exe（Station Viewer文件）；

附件7——底座SolidWorks源文件。